Towards Byzantine-resilient Learning in Decentralized Systems

Shangwei Guo[1], Tianwei Zhang[1], Xiaofei Xie[1], Lei Ma[2], Tao Xiang[3], and Yang Liu[1]

[1]*School of Computer Science and Engineering, Nanyang Technological University, Singapore*
[2]*Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan*
[3]*College of Computer Science, Chongqing University, Chongqing, China*

Abstract—With the proliferation of IoT and edge computing, decentralized learning is becoming more promising. When designing a distributed learning system, one major challenge to consider is Byzantine Fault Tolerance (BFT). Past works have researched Byzantine-resilient solutions for centralized distributed learning. However, there are currently no satisfactory solutions with strong efficiency and security in decentralized systems. In this paper, we propose a novel algorithm, MOZI, to achieve BFT in decentralized learning systems. Specifically, MOZI provides a uniform Byzantine-resilient aggregation rule for benign nodes to select the useful parameter updates and filter out the malicious ones in each training iteration. It guarantees that each benign node in a decentralized system can train a correct model under very strong Byzantine attacks with an arbitrary number of faulty nodes. We perform the theoretical analysis to prove the uniform convergence of our proposed algorithm. Experimental evaluations demonstrate the high security and efficiency of MOZI compared to all existing solutions.

I. INTRODUCTION

The rapid development of edge computing and deep learning technologies leads to the era of Artificial Intelligence of Things. Nowadays, it is a trend to learn and update powerful Deep Learning (DL) models on edge devices [1]–[5]. In contrast to the traditional approaches where training data and tasks are outsourced to third-party cloud platforms, model training in edge computing provides greater reliability and privacy protection. One typical example is the federated learning system [6], [7] where multiple edge devices collaborate to train a shared DL model.

However, federated learning introduces a centralized parameter server, which can bring new security and efficiency drawbacks [7]–[9]. First, federated learning suffers from single point of failure. The functionality of the system highly depends on the operations of the parameter server. If the server gets crashed or hacked, then the entire system will be broken down, affecting all the edge devices. Second, the centralized parameter server can be the performance bottleneck, particularly when a large amount of edge devices are connected to this server.

Due to the limitations of federated learning, there is a growing trend towards training a DL model in a decentralized fashion [8], [10]–[12]. Specifically, each node (e.g., edge device) in the decentralized network plays an euqal role (both training and aggregating parameters) in learning the model [13], [14]. This decentralization mode exhibits promising applications in many scenarios: in the Internet of Vehicles [15], cars can collaboratively learn powerful models of traffic sign classification for autonomous driving; in a social network, users can learn better recommendation models by communicating with their friends [16].

A distributed system can be threatened by the famous Byzantine Generals Problem [17]: a portion of nodes inside the network can conduct inappropriate behaviors, and propagate wrong information, leading to the failure of the entire system. This is particularly dangerous in both centralized and decentralized distributed learning scenarios. Distributed learning requires the collaboration of thousands of edge devices from different domains and parties. It is impossible to guarantee that each device is trusted and reliable. A single dishonest node can send wrong parameters/estimates to affect the entire network and final results. Past works have focused on the centralized distributed and federated learning with Byzantine-resilient Stochastic Gradient Descent (SGD) solutions [18]–[23]. The essential task for the parameter server is to distinguish between benign and malicious gradients and select the potential benign ones for model update. Based on the selection criteria, current solutions will fall into one of the two categories: (1) *distance*-based defense: the parameter server selects the gradients closer to the mean or median values in terms of vector distance [18]–[21], [24]; (2) *performance*-based defense: the server measures the model performance of each gradient using an extra validation dataset and chooses the ones with better performance [22], [25].

However, there are very few studies on BFT in decentralized learning systems. It is challenging to apply the above solutions to the decentralized scenario, due to two reasons. First, existing defenses themselves have security and efficiency drawbacks for protecting centralized systems. It has been shown that distance-based defenses are vulnerable to elaborately designed Byzantine attacks [25], [26]. Performance-based defenses have large computation overhead and scalability issue when conducting performance evaluation for each update. Besides, they require the parameter server to have an extra validation dataset, which is not realistic. These limitations still exist if the defenses were extended into decentralized systems. The second reason lies in the huge differences between centralized and decentralized learning systems. Existing defenses are mainly designed for one parameter server. However, each device in decentralized learning acts as not only a worker node, but

also a "parameter server". In addition, the number of neighbors connected to each device varies dramatically. So some assumptions made in the centralized defenses will not hold. To the best of our knowledge, currently there are only two research papers [27], [28] attempting to achieve Byzantine-resilient decentralized learning by adopting the *distance*-based strategy, which is vulnerable to sophisticated Byzantine attacks, as demonstrated in the evaluation section of this paper.

In this paper, we propose MOZI[1], a novel and efficient Byzantine-resilient algorithm for decentralized learning systems. MOZI has several innovations. (1) MOZI integrates both distance-based and performance-based strategies to detect Byzantine parameters. The distance-based strategy is used to reduce the scope of candidate nodes for better performance, and the performance-based strategy is used to further remove all the abnormal nodes for strong Byzantine Fault Tolerance. (2) In the distance-based selection phase, each benign node adopts its own parameters as the baseline instead of the mean or medium value of received untrusted parameters. This can enhance the quality of selected nodes, and defeat the sophisticated Byzantine attacks which attempt to manipulate the baseline value [19]. (3) In the performance-based selection phase, each benign node uses its training samples to test the performance of the selected parameters. This can perfectly relax the unrealistic assumption of the server' availability of validation datasets in centralized systems in performance-based strategies.

We theoretically analyze our proposed algorithm, and prove that the learned estimates of all nodes in the decentralized system can be uniformly converged to an ideal one. Besides, the convergence of decentralized systems still holds even with the presence of strong Byzantine attacks, i.e., an arbitrary number of Byzantine nodes in the case of satisfying the connectivity assumption. We conduct comprehensive experiments to show that MOZI is tolerant against both simple and sophisticated malicious behaviors, while all existing defense solutions fail. MOZI also achieves 8-30X performance improvement over existing popular defenses.

The key contributions of this paper are:

- We formally define the Byzantine Generals Problem in decentralized learning and analyze the vulnerabilities.
- We propose MOZI, a uniform Byzantine-resilient aggregation rule, to defeat an arbitrary number of Byzantine nodes in decentralized systems.
- We present theoretic analysis to prove the computational complexity and the convergence of MOZI.
- We conduct extensive experiments to show MOZI outperforms other solutions for both security and performance.

The rest of this paper is organized as follows. Related works are reviewed in Section II. Section III gives formal definitions of Byzantine General Problem in decentralized systems, and analyzes the vulnerabilities. Section IV presents our solution, MOZI, followed by the theoretical analysis. Section V shows

---

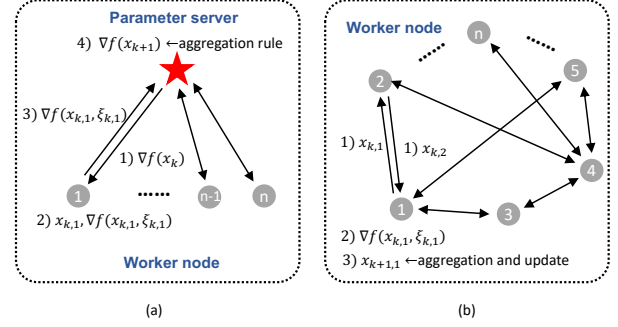[1]MOZI is the name of one famous ancient Chinese book on defensive warfare.

Fig. 1. Distributed learning systems in centralized (left) and decentralized (right) fashions.

the experimental results under various attacks and system settings. Section VI concludes the paper.

II. BACKGROUND AND RELATED WORK

A. Byzantine-resilient Centralized Learning

A centralized distributed learning system consists of a parameter server and multiple distributed worker nodes, as shown in Figure 1 (a). Every worker node has its own training dataset, but adopts the same training algorithm. In each iteration, a worker node 1) pulls the gradient from the parameter server and 2) calculates the gradient based on its local data. Then, the worker node 3) uploads the new gradient to the parameter server. After receiving all gradients from the worker nodes, the server 4) aggregates them into one gradient vector. The nodes repeat the above steps from the new gradient, until the training process is terminated and a model is produced.

Dishonest worker nodes in the system can compromise the training process and the final model by uploading wrong gradients [20], [25], [26]. It is necessary for the parameter server to detect such Byzantine nodes and discard their updates when aggregating the gradients. Existing defenses can be categorized into two classes, as described below.

Distance-based solutions. This type of solutions clusters the uploaded gradients and detects the outliers based on the vector distances. For instance, Blanchard et al. proposed Krum [18], which chooses the gradient vector with the minimal sum of squared distances to its neighbors as the aggregated one. Xie et al. designed Median-based Aggregation rules [19], which inspect the gradient vectors and calculate the median values in each dimension to defeat Byzantine attacks. Mhamdi et al. introduced Bulyan [20], to further enhance existing Byzantine-resilient aggregation rules by combining Krum and Median-based aggregation rules. Although these defenses can defeat simple Byzantine attacks such as Gaussian and bit-flip attacks [18], [19], they were shown to be vulnerable against more sophisticated attacks [25], [26]. The reason lies in the vulnerability of distance-based strategies: the close distance between two gradients does not imply similar performance.

Thus, these sophisticated attacks could create gradients that are malicious but indistinguishable from benign gradients in distance.

Performance-based Solutions. This type of solutions selects the benign nodes by evaluating the performance of each uploaded gradient. For instance, Xie et al. proposed Zeno [22] and Zeno++ [29] for synchronous and asynchronous learning systems, respectively. Both Zeno and Zeno++ calculate the prediction accuracy of each gradient on the extra validation datasets to identify Byzantine nodes. However, they require that the parameter server has a validation dataset, which is not realistic under some circumstances. Besides, performance evaluations of all gradients have much more overhead than parameter evaluations. This can significantly increase the total training time and the computation burdens for the parameter server, especially when the number of worker nodes is larger.

B. Byzantine-resilient Decentralized Learning

Decentralized learning systems remove the parameter server, as every node in the network is also responsible for model update. The architecture of a decentralized learning system is illustrated in Figure 1 (b). Specifically, in each iteration of the training process, each worker node 1) broadcasts its parameter vectors (estimates[2]) to its neighbor nodes, and receive the estimates from them; 2) trains the model estimates using the local data. 3) It then aggregates them with the neighbor nodes' estimates and updates the model.

Compared to centralized learning, research of Byzantine-resilient decentralized learning is still at an early stage. Yang et al. proposed ByRDiE [27] and BRIDGE [28], which simply apply the trimmed-median algorithm from centralized systems [19] to decentralized systems. While ByRDiE is designed for the coordinate descent optimization algorithm, BRIDGE is used in decentralized learning systems with SGD. Those two solutions follow the distance-based design strategy, and thus are vulnerable to some Byzantine attacks [20], [26]. In Section V, we will demonstrate their incapability of defeating sophisticated Byzantine attacks.

III. PROBLEM STATEMENT

In this section, we formally define a decentralized learning system under the Byzantine Fault circumstance and illustrate the vulnerability as well as the attack consequences.

A. Decentralized Systems

A decentralized system is defined as an undirected graph: $\mathcal{G} = (V, E)$, where V denotes a set of $\tilde{n}$ nodes and E denotes a set of edges representing communication links. Specifically, we have

- $(i, j) \in E$ if and only if node i can receive information from node j;
- $(j, i) \in E$ if $(i, j) \in E$.

Let $\mathcal{N}_i = \{j | (i, j) \in E\}$ be the set of the neighbors of node i. We further assume that n out of $\tilde{n}$ nodes are benign

---

[2]"Parameter vector" and "estimate" are used interchangeably.

and the rest are malicious. We can define a subgraph that only contains the benign nodes:

Definition 1. *(Benign Induced Subgraph) The benign induced subgraph, $\mathcal{G}_b = (V_b, E_b)$, is a subgraph of $\mathcal{G}$, formed by all the benign nodes in $\mathcal{G}$ and all the edges connecting those benign nodes. Specifically,*

- $i \in V_b \subseteq V$ *if i is a benign node and $|V_b| = n$;*
- $(i, j) \in E_b \subseteq E$ *if and only if $i, j \in V_b$;*
- $(j, i) \in E_b$ *if $(i, j) \in E_b$.*

Following the information exchange models in [13], [14], we assume the benign induced subgraph is fully connected, i.e., giving two arbitrary benign nodes i and j, there always exists at least one path that connects these two nodes. We formally state the assumption as below:

Assumption 1. *(Connectivity of Benign Induced Subgraph) There exists an integer τ such that for $\forall i, j \in V_b$, node j can propagate its information to node i through at most τ edges.*

B. Model Training under Byzantine Fault

In the decentralized learning system, $\tilde{n}$ nodes cooperatively train a model by optimizing the loss function with SGD and exchanging estimates with their neighbors, while Byzantine nodes attempt to tamper with such training process. Let $x \in \mathbb{R}^d$ be the d-dimensional estimate vector of a DL model; l be the loss function. Each node $i \in V_b$ obtains a training dataset D_i, consisting of independent and identically distributed (i.i.d.) data samples from a distribution D. Those $\hat{n}$ nodes train a shared model by solving the following optimization problem with the presence of Byzantine nodes:

$$\min_{x \in \mathbb{R}^d} \mathbb{E}_{\xi \sim D} l(x; \xi)$$

where ξ is a training data sample from D.

At the k-th iteration, node i has its local estimate denoted as $x_{k,i}$, and broadcasts it to its neighbors. When receiving the estimates from the neighbors, node i will update its local estimate according to the General Update Function (GUF):

Definition 2. *(GUF) Let $x_{k,i}$, $\nabla l(x_{k,i}, \xi_{k,i})$ be the estimate and gradient of node i in the k-th iteration. $\{x_{k,j}, j \in \mathcal{N}_i\}$ are the estimates from its neighbors. $\mathcal{R}$ is an aggregation rule. Node i updates its estimate for the $(k + 1)$-th iteration using the following general update function:*

$$x_{k+1,i} = \alpha x_{k,i} + (1 - \alpha)\mathcal{R}(x_{k,j}, j \in \mathcal{N}_i) - \lambda \nabla l(x_{k,i}, \xi_{k,i}) \quad (1)$$

where λ is the learning rate; α is a hyper-parameter that balances the weights of the estimates.

Without loss of generality, we assume all the nodes have the same learning rate. The stochastic gradient can be replaced with a mini-batch of stochastic gradients [8].

A straightforward way is the average aggregation rule:

$$\mathcal{R}_{Average} = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} x_{k,j} \quad (2)$$

and α is set as $\frac{1}{|\mathcal{N}_i|+1}$. However, because the average aggregation in Equation 2 does not consider BFT, this training process can be easily compromised by Byzantine attacks: an adversary can use just one malicious node to send wrong estimates to its neighbors and alter their aggregated estimates. More seriously, due to the fully connectivity of benign induced subgraph, this fault will be also propagated to other nodes not directly connected to this Byzantine node after several iterations, and finally all the nodes in this network will be affected.

Lemma 1. *Consider a decentralized system, in which $\hat{i}$ is a Byzantine node, attempting to add a malicious vector $\hat{x}$ to a benign node $i_{\tau'}$'s estimate. The shortest distance (i.e., number of edges) between them is τ' and $\{i_s\}_{s=1}^{\tau'-1}$ are the benign nodes on the shortest trace between $i_{\tau'}$ and $\hat{i}$. The distance between node i_s and $\hat{i}$ is s. Then at k_0-th iteration, node $\hat{x}$ can broadcast to its neighbors the following estimate to achieve this goal in τ' iterations:*

$$x = x_{k_0,\hat{i}} + \hat{x} \prod_{s=1}^{\tau'} (|\mathcal{N}_{i_s}| + 1) \qquad (3)$$

where $|\mathcal{N}_{i_s}|$ is the number of neighbors of node i_s.

Proof. See Appendix A $\qquad\qquad\qquad\qquad\qquad$ $\square$

C. Byzantine-resilient Aggregation Rule

It is necessary to design a robust aggregation rule in a decentralized system to defeat Byzantine nodes. This rule should guarantee that all benign nodes converge to the optimal estimate learned without Byzantine nodes. We define the Byzantine-resilient aggregation rule as below:

Definition 3. *(Uniform Byzantine-resilience) Consider a decentralized system $\mathcal{G}$ under Byzantine Fault. Let x^* be the optimal model. An aggregation rule $\mathcal{R}$,*

$$\mathcal{R} = \mathcal{R}(x_j \text{ for } j \in \mathcal{N}_i),$$

is said to be uniform Byzantine-resilient if, for all benign nodes, the distances between their estimates and x^* is uniformly bounded. Formally, there exists a constant C^* such that for $\forall i \in V_b$, $k \to \infty$,

$$\|x_{k,i} - x^*\| \le C^*. \qquad (4)$$

D. Existing Byzantine Defenses

The aggregation rule must be designed to be secure against Byzantine attacks, and also guarantee convergence. We can extend existing aggregation rules from centralized systems to the decentralized case, as the estimates and gradients have the same dimensionality and network structure. We consider three popular Byzantine-resilient solutions: Krum [18], marginal median [19], and Bulyan [20]. Other solutions can be extended in similar ways.

It is worth noting that in a centralized distributed system, the maximal number of Byzantine workder nodes connected to the parameter server is usually assumed. This does not hold in a decentralized system, as the number of Byzantine nodes connected to each benign node varies greatly. To meet this assumption, we approximately calculate the number of Byzantine nodes allowed for each benign node in Equation 5. In this equation, $\hat{n}_i$ is the number of Byzantine nodes connected to node i, $\rho_{central}$ is the maximal ratio of Byzantine nodes allowed in a centralized distributed defense, and $\lceil \cdot \rceil$ is the ceil function.

$$\hat{n}_i = \lceil |\mathcal{N}_i| \cdot \min\{1 - \rho_i, \rho_{central}\} \rceil \qquad (5)$$

DKrum. Similar to Krum, let $j \ne j'$ be two neighbors of node i and we denote $j \to j'$ the $|\mathcal{N}_i| - \hat{n}_i + 2$ closest estimate vectors to the estimate of node j. Then, we calculate the score of each neighbor:

$$s(j) = \sum_{j \to j'} \|x_j - x_{j'}\|^2$$

We select the estimate with the minimal score as the aggregated estimate. Formally, the aggregated rule DKrum is defined as

$$\mathcal{R}_{DKrum} = x_{j^*}, \ j^* = \underset{j \in \mathcal{N}_i}{\operatorname{argmin}} \, s(j).$$

Dmedian. To apply the marginal median solution [19] to decentralized systems, we only need to replace the gradients with the received estimates. Specifically, the aggregated rule $Dmedian$ is defined as

$$\mathcal{R}_{Dmedian} = \texttt{MarMed}\{x_j, j \in \mathcal{N}_i\} \qquad (6)$$

where $\texttt{MarMed}$ is the marginal median function defined in [19]: the m-th dimensional value of $\mathcal{R}_{Dmedian}$ is the median of the m-th dimensional elements of all estimates in $\mathcal{N}_i$.

DBulyan. At each iteration, node i first recursively uses DKrum to select $|\mathcal{N}_i| - 2\hat{n}_i$ estimates, i.e., $\{x_j, j \in \mathcal{N}_i^{DKrum}\}$, where $|\mathcal{N}_i^{DKrum}| = |\mathcal{N}_i| - 2\hat{n}_i$. Specifically, node i uses DKrum to select one estimate from its neighbors and deletes the corresponding node from its neighbors. Then, node i recursively selects the remaining estimates using DKrum. Finally, it adopts a median-based method to aggregate the estimates in $\{x_j, j \in \mathcal{N}_i^{DKrum}\}$. Formally, the m-th coordinate of the aggregated estimate is calculated as

$$\mathcal{R}_{DBulyan}[m] = \frac{1}{\beta} \sum_{j \in \mathcal{M}[m]} x_j[m] \qquad (7)$$

where $\beta = |\mathcal{N}_i| - 4\hat{n}_i$ and $\mathcal{M}[m]$ is the set of neighbors with the size of β. The sum of the m-th elements to its median is minimal among all subsets of $\mathcal{N}_i^{DKrum}$ with the size β.

IV. ALGORITHM

A. MOZI

As we introduced in Section II, existing Byzantine defenses for centralized systems have certain vulnerabilities or limitations. Such design flaws still exist when we extend the solutions to decentralized scenarios. Besides, a decentralized

system has higher convergence requirement than a centralized system: convergence of one parameter server enforced by the solutions cannot guarantee the convergence of all benign nodes in a decentralized system. Further, centralized Byzantine-resilient solutions usually assume a fixed number of faulty nodes connected to the parameter server, while in a decentralized system, the number of faulty nodes connected to each benign node varies significantly. As such, it is necessary to have a more robust Byzantine-resilient solution that can defeat *an arbitrary number* of Byzantine nodes and guarantee convergence of *each benign node* in decentralized systems.

We propose MOZI to achieve this goal. Our solution consists of two stages for each training iteration. At the first stage, each benign node performs a distance-based strategy to select a candidate pool of potential benign nodes from its neighbors. The selection is made by comparing the Euclidean distance of the estimate of each neighbor node with its own estimate. Although after this stage, the candidate pool might still contain Byzantine nodes as the distance-based strategies are not strict Byzantine-resilient, it indeed reduces the scope of benign nodes for further selection. At the second stage, each benign node performs a performance-based strategy to pick the final nodes from the candidate pool for estimate update. It reuses the training sample as the validation data to test the performance (i.e., loss function value) of each estimate. It selects the estimates whose loss values are smaller than the value of its own estimate, and calculates the average of those estimates as the final updated value. Although such performance-based strategy can incur certain computation cost, the small candidate pool can significantly reduce the impact compared to previous works. MOZI can be formally described as below:

Definition 4. *(MOZI) Let $x_{k,i}$ be the k-th estimate of node i; $l_{k,i}$ be the loss of the estimates on the k-th stochastically selected data sample, i.e., $l_{k,i} = l(x_{k,i}, \xi_{k,i})$; ρ_i be the ratio of benign neighbors of node i. The proposed uniform Byzantine resilient aggregation rule, MOZI, is define as*

$$\mathcal{R}_{\text{MOZI}} = \begin{cases} \frac{1}{|\mathcal{N}_{k,i}^r|} \sum_{j \in \mathcal{N}_{k,i}^r} x_{k,j}, & \text{if } \mathcal{N}_{k,i}^r \neq \emptyset \\ x_{k,j^*}, & \text{Otherwise} \end{cases} \quad (8)$$

where

$$\text{(Stage 1) } \mathcal{N}_{k,i}^s = \underset{\substack{\mathcal{N}^* \subseteq \mathcal{N}_i \\ |\mathcal{N}^*| = \rho_i |\mathcal{N}_i|}}{\operatorname{argmin}} \sum_{j \in \mathcal{N}^*} ||x_{k,j} - x_{k,i}||,$$

$$\text{(Stage 2) } \mathcal{N}_{k,i}^r = \bigcup_{\substack{j \in \mathcal{N}_{k,i}^s \\ l_{k,j} \leq l_{k,i}}} j, \text{ and } j^* = \underset{j \in \mathcal{N}_{k,i}^s}{\operatorname{argmin}} l_{k,j}.$$

Algorithm 1 details the training process of node i using MOZI in a decentralized system. The algorithm begins with the estimate $x_{0,i} = x_0$. At the k-th iteration, node i broadcasts its estimate to and receives the estimates from its neighbors. It stochastically selects a training data sample $\xi_{k,i}$ and calculates the loss and the gradient (Lines 4-6). Then it conducts two-stage estimate selection. First, it calculates the Euclidean distances between $x_{k,i}$ and the estimates from its neighbors

---

Algorithm 1: The training algorithm for each node in MOZI.

1: **Input:** initial estimate x_0, learning rate λ, number of iterations K, ratio of benign nodes ρ_i, α
2: **for** k in $[0, K)$ **do**
3:     Broadcast $x_{k,i}$ and receive $x_{k,j}$ from $j \in \mathcal{N}_i$
4:     Stochastically sample $\xi_{k,i}$ from D_i
5:     $l_{k,i} \leftarrow l(x_{k,i}, \xi_{k,i})$
6:     $\bigtriangledown l(x_{k,i}, \xi_{k,i}) \leftarrow$ Compute the local gradient
7:     **for** j in $\mathcal{N}_i$ **do**
8:         $d_{i,j} \leftarrow ||x_{k,i} - x_{k,j}||$
9:     **end for**
10:    $\mathcal{N}_{k,i}^s \leftarrow \operatorname{argmin}_{\substack{\mathcal{N}^* \subseteq \mathcal{N}_i \\ |\mathcal{N}^*| = \rho_i |\mathcal{N}_i|}} \sum_{j \in \mathcal{N}^*} d_{i,j}$
11:    **for** $j \in \mathcal{N}_{k,i}^s$ **do**
12:        $l_{k,j} \leftarrow l(x_{k,j}, \xi_{k,i})$
13:        **if** $l_{k,i} - l_{k,j} \geq 0$ **then**
14:            $\mathcal{N}_{k,i}^r \leftarrow j$
15:        **end if**
16:    **end for**
17:    **if** $\mathcal{N}_{k,i}^r$ *is* $\emptyset$ **then**
18:        $\mathcal{N}_{k,i}^r \leftarrow \operatorname{argmin}_{j \in \mathcal{N}_{k,i}^s} l_{k,j}$
19:    **end if**
20:    $\mathcal{R}_{k,i} \leftarrow \frac{1}{|\mathcal{N}_{k,i}^r|} \sum_{j \in \mathcal{N}_{k,i}^r} x_{k,j}$
21:    Update the local estimate
        $x_{k+1,i} \leftarrow \alpha x_{k,i} + (1 - \alpha)\mathcal{R}_{k,i} - \lambda \bigtriangledown l(x_{k,i}, \xi_{k,i})$
22: **end for**
23: **return** $x_{K,i}$

---

and selects $\rho_i |\mathcal{N}_i|$ neighbors with lowest distances (Lines 7-10).

Second, for each estimate $x_{k,j}, j \in \mathcal{N}_{k,i}^s$, node i calculates the loss of $x_{k,j}$ on $\xi_{k,i}$. It chooses the estimates that have similar or better performance than that of $x_{k,i}$ (Lines 11-19). Finally it calculates the average value of the selected nodes and updates the final estimate using GUF (Lines 20-21).

B. Complexity Analysis

The training process with MOZI is performance efficient, as proved below:

Proposition 1. *(Cost of MOZI) The computational complexity of MOZI is $O(|\mathcal{N}_i|d)$ for each node at each iteration.*

Proof. For node $i \in V_b$, at each iteration, MOZI aggregates the received estimates with three operations. First, MOZI selects the $\rho_i |\mathcal{N}_i|$ neighbors that are closest to its current estimate. The cost is $O(|\mathcal{N}_i|d)$. Second, MOZI calculates the loss of the selected estimates on the stochastic sample and the cost is $O(\rho_i |\mathcal{N}_i|d)$. Finally, MOZI takes at most $O(\rho_i |\mathcal{N}_i|d)$ to aggregate the estimates with better performance. Since $\rho_i \leq 1$ for $\forall i \in V_b$, the overall computational complexity of MOZI is $O(|\mathcal{N}_i|d)$. $\square$

Compared to existing aggregation rules, the complexity of Average aggregation, Median-based and BRIDGE is $O(|\mathcal{N}_i|d)$,

while Krum and Bulyan have a complexity of $O(|\mathcal{N}_i|^2 d)$. So we conclude that MOZI maintains the same performance efficiency as some solutions and performs much better than others, especially when the number of connected neighbor nodes becomes large.

C. Convergence Analysis

We adopt the following assumptions for our convergence analysis, which are also commonly used in [8], [14], [22].

Assumption 2. *We assume that all worker nodes are initialized with the same model parameter x_0.*

Since Byzantine nodes are trying to compromise the training procedure, we have the following assumption.

Assumption 3. *(Monotonic Performance) We assume that the performance of benign estimates is higher than the Byzantine estimates on the predefined training distribution D at each iteration. Specifically, let $x_k, \hat{x}_k$ be a benign and Byzantine estimates at the k-th iteration. Then,*

$$\mathbb{E}_{\xi \sim D} l(x_k, \xi) > \mathbb{E}_{\xi \sim D} l(\hat{x}_k, \xi) \quad (9)$$

To ensure the convergence of decentralized systems without Byzantine nodes, we also assume the bounded variance of estimates and gradients.

Assumption 4. *(Bounded Variance) We assume that the difference between all gradients and estimates generated by benign nodes are uniformly bounded. Specifically, there exists scalars T and T' such that, for $\forall i, j \in V_b$,*

$$\max_{i,j \in V_b} \{ ||x_{k,i} - x_{k,j}|| \} \leq T \quad (10)$$

$$\max_{i,j \in V_b} \{ || \bigtriangledown l(x_{k,i}) - \bigtriangledown l(x_{k,j}) || \} \leq T' \quad (11)$$

We assume that the loss function l is a convex function and Lipschitz continuity. With Assumption 1-4 and the assumption on the loss function, we present the convergence of a benign induced subgraph, which has been proven in [8], [14].

Lemma 2. *(Convergence of Benign Induced Subgraph) We assume that the sub-decentralized system, $\mathcal{G}_b$, uniformly converges. Specifically, let x^* be the optimal model. For $\forall i \in V_b$, the estimate and loss differences with the optimal ones are uniformly bounded by predefined constants C and C', i.e.,*

$$||x_{k,i} - x^*|| \leq C$$
$$l(x_{k,i}) < l(x^*) + C'$$

Let $\mathcal{G}$ be a decentralized system with Byzantine nodes. It follows the Byzantine-resilient strategy in Algorithm 1 to train a model. Under the given assumptions, we show that the differences among the estimates of all benign nodes are uniformly bounded.

Lemma 3. *(Consistency of Benign Nodes) Consider the above decentralized system $\mathcal{G}$ with Assumptions 1-4. We claim that for $\forall i, j \in V_b$, $k \to \infty$,*

$$||x_{k,i} - x_{k,j}|| \leq 3T \frac{1 - \alpha^{k-1}}{1 - \alpha} + \lambda T' \frac{1 - \alpha^{k-1}}{1 - \alpha} \quad (12)$$

Proof. See Appendix B □

Then, we prove that the proposed MOZI is uniform Byzantine resilience.

Theorem 1. *(Convergence of MOZI) Consider the above decentralized system $\mathcal{G}$ with Assumptions 1-4. Let x^* be the optimal estimate. For $\forall$ benign nodes $i, j \in \mathcal{G}$, we have*

$$||x_{k,i} - x^*|| \leq 2T \frac{1 - \alpha^{k-1}}{1 - \alpha} + \lambda T' \frac{1 - \alpha^{k-1}}{1 - \alpha} + C, \quad (13)$$

where C is a predefined constant.

Proof. See Appendix C □

We also show that the differences between the losses of each benign node and the optimal are uniformly bounded.

Corollary 1. *(Convergence of Loss) Consider the above decentralized system $\mathcal{G}$ with Assumption 1-4. Let x^* be the optimal estimate. For $\forall$ benign nodes $i, j \in \mathcal{G}$, we have*

$$l(x_{k,i}) - l(x^*) \leq C', \quad (14)$$

where C is a predefined constant.

Proof. See Appendix D □

Note that Theorem 1 and Corollary 1 have no restrictions on the ratio of Byzantine nodes connected to each benign node. Therefore, the proposed MOZI is Byzantine tolerant against an arbitrary number of faulty nods, only if the connectivity of benign induced subgraph (Assumption 1) is satisfied.

V. EXPERIMENTS

A. Experimental Setup and Configurations

Datasets. We evaluate our defense solution with a DL-based image classification task. Specifically, we train a Convolutional Neural Network (CNN) over the CIFAR10 dataset [30]. This CNN includes two maxpooling layers and three fully connected layers [20]. We adopt a batch size of 256 and a fading learning rate λ with the initial learning rate 0.05. We also conduct the same evaluations on a simpler MNIST dataset. The experimental results lead to the same conclusion, and can be found in Appendix E.

Implementation of decentralized systems. The network topology of the decentralized system in our consideration is defined by a connection rate between the nodes. A connection rate is the probability that a node is connected to another node. To ensure the connectivity assumption, we first generate a decentralized network in which all the nodes strictly follow the learning procedure. Then we randomly add Byzantine nodes to the network. To simulate various adversarial environments, we adopt a new parameter, Byzantine ratio, which is defined as the number of Byzantine nodes divided by the number of all nodes in the network. We assume that the Byzantine ratio of node i is lower than $1 - \rho_i$. Without lose of generality, we set ρ_i as 0.4 for all benign nodes. we set ρ_i as 0.4. We train the deep learning model in a synchronous mode. We simulate the operations of the decentralized system by running the nodes
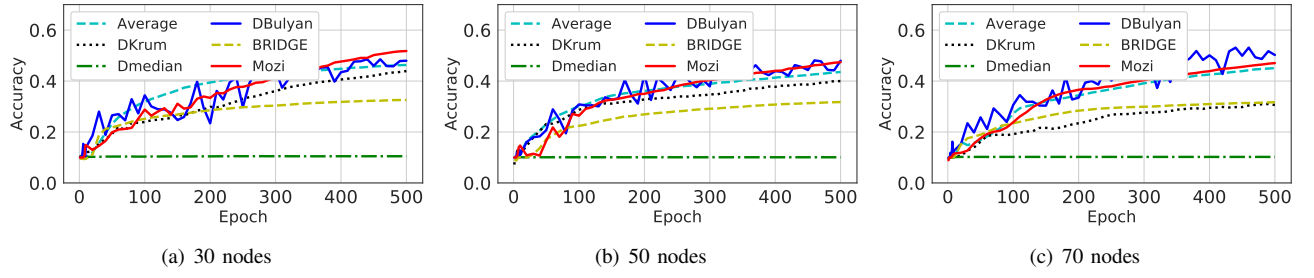
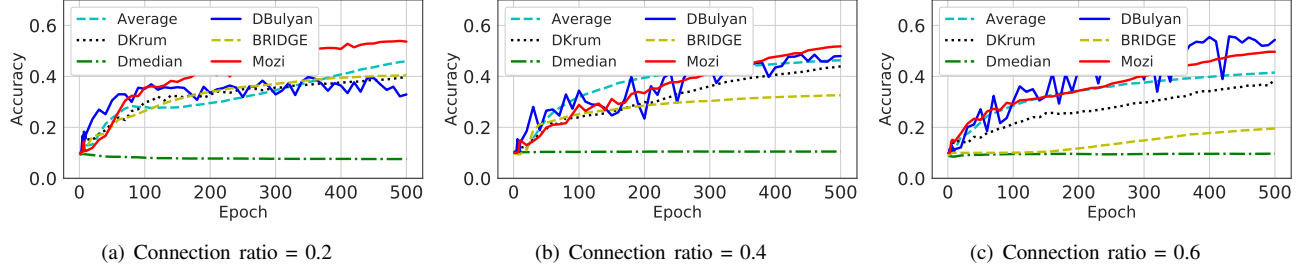Fig. 2. The worst accuracy of the benign nodes with different network sizes.

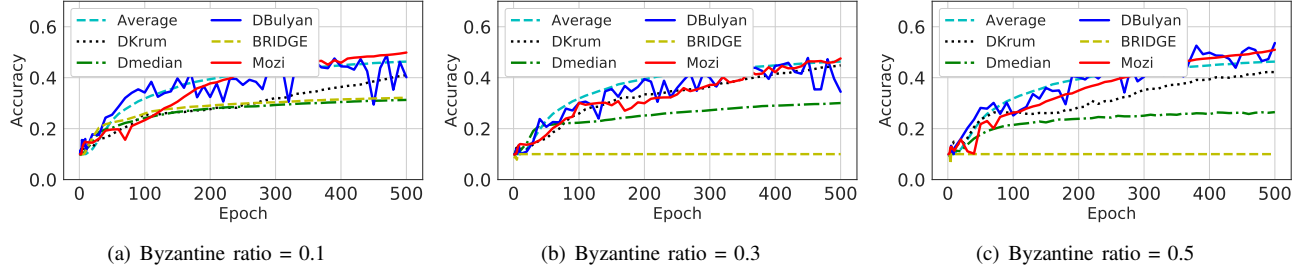Fig. 3. The worst accuracy of the benign nodes with different connection ratios.

Fig. 4. The worst accuracy of the benign nodes under the Gaussian attack.
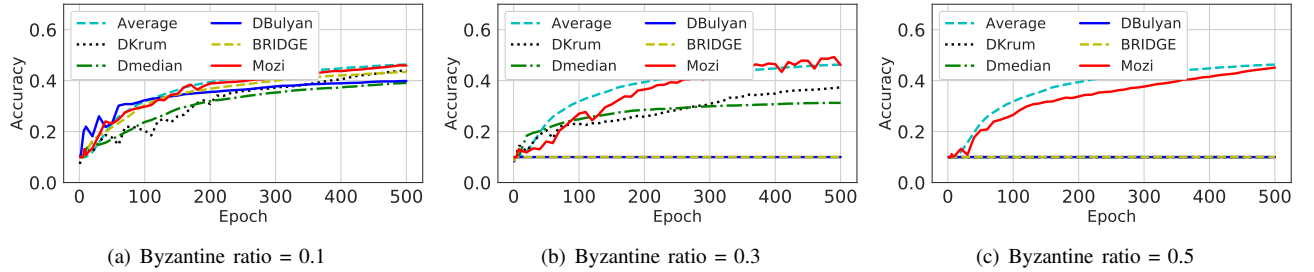

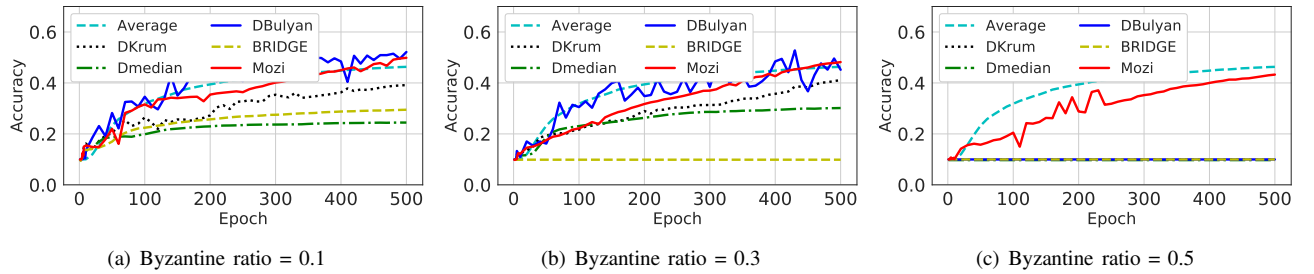Fig. 5. The worst accuracy of the benign nodes under the bit-flip attack.

Fig. 6. The worst accuracy of the benign nodes under the Mhamdi attack.

serially at each iteration. All our experiments are conducted on a server equipped with Xeon Silver 4214 @2.20 GHz CPUs and a NVIDIA Tesla P40 GPU.

Implementation of Byzantine-resilient defenses. In addition to our proposed MOZI, we also implement four other defenses for comparison: three are extended from existing popular defenses for centralized systems, i.e., DKrum, Dmedian and DBulyan and one is designed specifically for decentralized systems, i.e., BRIDGE [28]. For all these defenses, we set α to be 0.5 in the GUF. For baseline, we consider the same decentralized system configuration without Byzantine nodes, and using the Average Aggregation rule (Equation 2). The model trained from this setting can be regarded as the optimal one. For each defense deployed in the decentralized system, we measure the testing accuracy of the trained model on each benign node, and report the worst accuracy among all nodes to represent the effectiveness of this defense.

B. Convergence

As an aggregation rule in a decentralized system, the essential functionality is to achieve uniform convergence, i.e., the model in each benign node must be converged to the correct one. We evaluate the convergence functionality of MOZI with different configurations.

Network size. We first evaluate the convergence of our solution under different network sizes. It is more difficult to achieve uniform convergence when there are more nodes. In our experiments, we consider a decentralized system with 30, 40 and 50 nodes respectively [20]. The connection ratio is set as 0.4. Figure 2 shows the worst accuracy during the training phase on CIFAR10.

We can observe that only DKrum, DBulyan and our proposed MOZI have the same convergence as the baseline. MOZI converges to a slightly better model at a higher speed. In contrast, BRIDGE and Dmedian have bad convergence performance, especially when the network size is larger.

Network connection ratio. This factor can also affect the model convergence: it takes more effort and time for all nodes to reach the consensus when the connection is heavier. We evaluate such impact with different connection ratio (0.2, 0.4 and 0.6), while fixing the number of nodes as 30.

Figure 3 illustrates that most defense solutions in our consideration have satisfactory convergence performance when the connection ratio is small (0.2 and 0.4). Our proposed MOZI has better convergence performance when the connection ratio is 0.6. The BRIDGE and DMedian approaches cannot produce correct models at this high connectivity.

C. Byzantine Fault Tolerance

We evaluate the performance of different defense strategies under various Byzantine attacks. We set the connection ratio of the evaluated system as 0.4 and the number of benign nodes as 30. We consider different Byzantine ratios (0.1, 0.3 and 0.5).

Guassian attack. We first use a simple attack to test the Byzantine resilience. Specifically, in each iteration the adversarial nodes broadcast to their neighbors random estimate

vectors following the Gaussian distribution. We refer to this kind of attack as Gaussian attack.

Figure 4 shows the model training performance under the Gaussian attack. The advantage of MOZI over other strategies is obvious. Dmedian, DBulyan and BRIDGE do not uniformly converge in all systems. DKrum fails to converge at the Byzantine ratio of 0.5. Only MOZI can generate the correct model regardless of the Byzantine ratio.

Bit-flip attack. We also implement a bit-flip attack [19] to evaluate these defenses, where the adversarial nodes flip the sign of the floating estimates and broadcast them to their neighbors.

Figure 5 shows the convergence results: the advantage of MOZI over other strategies is more obvious. MOZI has the same performance as the baseline regardless of the Byzantine ratio. This indicates that it is absolutely Byzantine-resilient against the bit-flip attack. In contrast, DBulyan and BRIDGE do not converge uniformly when the Byzantine ratio is high (0.3 and 0.5). DKrum and Dmedian also fail to defeat the bit-flip attack when the Byzantine ratio is 0.5.

Sophisticated attack. To fully evaluate the BFT of our proposed approach, we adopt a more sophisticated attack, Mhamdi attack [20]: the adversary has the capability of collecting all the uploaded estimates from other neighbor nodes. Then it can carefully design its own estimate to make it undetectable from the benign ones, while still compromising the training process. Mhamdi attack has been shown effective against most defenses in centralized systems [20].

The results are shown in Figure. 6. We observe that MOZI can always succeed for different Byzantine ratios. In contrast, other solutions fail to defeat Mhamdi attack in some cases: all solutions fail to converge when the Byzantine ratio is 0.5. Dmedian and BRIDGE cannot converge even when the Byzantine ratio is 0.1.

D. Computation Cost

To evaluate the computation cost of MOZI, we measure the average training and aggregation time for one iteration on CIFAR10. We consider a decentralized system with 30 nodes and the connectivity is 0.4. Figure 7 shows the average time of different defense solutions. We can see that the training time for those solutions are identical, but the aggregation time differs a lot. MOZI can finish one iteration in a much shorter time than DKrum (8X faster) and DBulyan (30X faster). The reason is that while MOZI only calculates the difference between a node and its neighbors, DKrum and DBulyan have to compare the distances among all neighbors. MOZI is slightly worse than Dmedian and BRIDGE. But considering the bad convergence of Dmedian and BRIDGE under Byzantine attacks demonstrated in Section V-C, we conclude that MOZI is the optimal solution with the strongest Byzantine resilience and acceptable computation overhead.

VI. CONCLUSION

In this paper, we explore the Byzantine Fault Tolerance in decentralized learning systems. We formally define the Byzan-
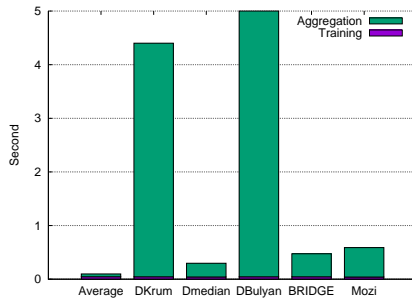
Fig. 7. The average training and aggregation time of one iteration for different aggregation rules.

tine General Problem and demonstrate that a decentralized system is highly vulnerable to Byzantine attacks. We show that existing Byzantine-resilient solutions in centralized systems cannot be used to protect decentralized systems due to their security flaws and inefficiency. Then we propose an uniform Byzantine-resilient approach, MOZI to defeat Byzantine attacks in decentralized learning. We conduct theoretic analysis to prove that MOZI has strong convergence. Experimental results reveal that MOZI can resist both simple and sophisticated Byzantine attacks with low computation overhead under different system configurations and training tasks.

REFERENCES

[1] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[2] T. Zhang, Y. Zhang, and R. B. Lee, "Analyzing cache side channels using deep neural networks," in *Proceedings of the 34th Annual Computer Security Applications Conference*, 2018, pp. 174–186.

[3] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: A survey," *IEEE Communications Surveys and Tutorials*, 2019.

[4] Y. Wang, Z. Jiang, X. Chen, P. Xu, Y. Zhao, Y. Lin, and Z. Wang, "E2-Train: Training state-of-the-art CNNs with over 80% energy savings," in *Advances in Neural Information Processing Systems*, 2019, pp. 5139–5151.

[5] J. Sun, T. Zhang, X. Xie, L. Ma, Y. Zheng, K. Chen, and Y. Liu, "Stealthy and efficient adversarial attacks against deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[6] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.

[7] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecny, S. Mazzocchi, H. B. McMahan *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.

[8] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 5330–5340.

[9] X. Xie, L. Ma, H. Wang, Y. Li, Y. Liu, and X. Li, "Diffchaser: Detecting disagreements for deep neural networks," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2019, pp. 5772–5778.

[10] R. Dobbe, D. Fridovich-Keil, and C. Tomlin, "Fully decentralized policies for multi-agent systems: An information theoretic approach," in *Advances in Neural Information Processing Systems*, 2017, pp. 2941–2950.

[11] H. Tang, S. Gan, C. Zhang, T. Zhang, and J. Liu, "Communication compression for decentralized training," in *Advances in Neural Information Processing Systems*, 2018, pp. 7652–7662.

[12] A. Lalitha, X. Wang, O. Kilinc, Y. Lu, T. Javidi, and F. Koushanfar, "Decentralized bayesian learning over graphs," *arXiv preprint arXiv:1905.10466*, 2019.

[13] J. N. Tsitsiklis, "Problems in decentralized decision making and computation." Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems, Tech. Rep., 1984.

[14] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, p. 48, 2009.

[15] A. Gulati, G. S. Aujla, R. Chaudhary, N. Kumar, and M. S. Obaidat, "Deep learning-based content centric data dissemination scheme for Internet of Vehicles," in *IEEE International Conference on Communications*, 2018, pp. 1–6.

[16] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi, "Personalized and private peer-to-peer machine learning," in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 473–481.

[17] L. LAMPORT, R. SHOSTAK, and M. PEASE, "The Byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982.

[18] P. Blanchard, R. Guerraoui, J. Stainer *et al.*, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 119–129.

[19] C. Xie, O. Koyejo, and I. Gupta, "Generalized Byzantine-tolerant SGD," *arXiv preprint arXiv:1802.10116*, 2018.

[20] E. M. E. Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *International Conference on Machine Learning*, 2018, pp. 3521–3530.

[21] L. Chen, H. Wang, Z. Charles, and D. Papailiopoulos, "DRACO: Byzantine-resilient distributed training via redundant gradients," in *International Conference on Machine Learning*, 2018, pp. 903–912.

[22] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *International Conference on Machine Learning*, 2019, pp. 6893–6901.

[23] ——, "SLSGD: Secure and efficient distributed on-device machine learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2019.

[24] G. Damaskinos, R. Guerraoui, R. Patra, M. Taziki *et al.*, "Asynchronous Byzantine machine learning (the case of SGD)," in *International Conference on Machine Learning*, 2018, pp. 1145–1154.

[25] M. Fang, X. Cao, J. Jia, and N. Z. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *USENIX Security Symposium*, 2020.

[26] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 8632–8642.

[27] Z. Yang and W. U. Bajwa, "ByRDiE: Byzantine-resilient distributed coordinate descent for decentralized learning," *IEEE Transactions on Signal and Information Processing over Networks*, 2019.

[28] ——, "BRIDGE: Byzantine-resilient decentralized gradient descent," *arXiv preprint arXiv:1908.08098*, 2019.

[29] C. Xie, O. Koyejo, and I. Gupta, "Zeno++: Robust fully asynchronous SGD," *arXiv preprint arXiv:1903.07020*, 2019.

[30] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

A. Proof of Lemma 1

Proof. We assume that there is only one path of τ' edges from node $\hat{i}$ to node $i_{\tau'}$. We show that the Byzantine node $\hat{i}$ only needs to send

$$\hat{x}_{k_0,\hat{i}} = x_{k_0,\hat{i}} + \hat{x} \prod_{s=1}^{\tau'} (|\mathcal{N}_{i_s}| + 1)$$

to its neighbors at the current iteration k_0. We prove the lemma by mathematical induction.

If $\tau' = 1$, the two nodes are neighbors. Then, the estimate of node i_1 at $k_0 + 1$ iteration is

$$\hat{x}_{k_0+1,i_1} = \frac{1}{|\mathcal{N}_{i_1}| + 1}(x_{k_0,i_1} + \sum_{j \in \mathcal{N}_i/\hat{i}} x_{k_0,j} + x_{k_0,\hat{i}} + \hat{x}(|\mathcal{N}_{i_1}| + 1)) - \lambda \bigtriangledown l(x_{k_0,i_1}, \xi_{k_0,i_1})$$

$$= \frac{1}{|\mathcal{N}_{i_1}| + 1}(x_{k_0,i_1} + \sum_{j \in \mathcal{N}_i} x_{k_0,j}) - \lambda \bigtriangledown l(x_{k_0,i_1}, \xi_{k_0,i_1}) + \hat{x}$$

It proves that the lemma is true when $\tau' = 1$.

We now assume the lemma is true when $\tau' = k$, i.e. node $\hat{i}$ sends

$$\hat{x}_{k_0,\hat{i}} = x_{k_0,\hat{i}} + \hat{x} \prod_{s=1}^{k} (|\mathcal{N}_{i_s}| + 1).$$

to its neighbors at k_0-th iteration. Then, at $(k_0 + k)$-th iteration, node i_k's estimate is

$$\hat{x}_{k_0+k,i_k} = x_{k_0+k,i_k} + \hat{x}$$

where x_{k_0+k,i_k} is the benign estimate if node $\hat{i}$ was not controlled.

Consider the $k + 1$ case. Node $\hat{i}$ sends

$$\hat{x}_{k_0,\hat{i}} = x_{k_0,\hat{i}} + \hat{x} \prod_{s=1}^{k+1} (|\mathcal{N}_{i_s}| + 1)$$

to its neighbors at k_0-th iteration. At $k_0 + k$-th iteration, node i_k's estimate is

$$\hat{x}_{k_0+k,i_k} = x_{k_0+k,i_k} + \hat{x}(|\mathcal{N}_{i_{k+1}}| + 1).$$

Then, at $(k_0 + k + 1)$-th iteration, the estimate of node x_{k+1} is

$$\hat{x}_{k_0+k+1,i_{k+1}} = \frac{1}{|\mathcal{N}_{i_{k+1}}| + 1}(x_{k_0+k,i_{k+1}} + \sum_{j \in \mathcal{N}_{i_{k+1}}/i_k} x_{k_0+k,j} + x_{k_0+k,i_k} + \hat{x}(|\mathcal{N}_{i_{k+1}}| + 1)) - \lambda \bigtriangledown l(x_{k_0+k,i_{k+1}}, \xi_{k_0+k,i_{k+1}})$$

$$= \frac{1}{|\mathcal{N}_{i_{k+1}}| + 1}(x_{k_0+k,i_{k+1}} + \sum_{j \in \mathcal{N}_{i_{k+1}}} x_{k_0+k,j}) - \lambda \bigtriangledown l(x_{k_0+k,i_{k+1}}, \xi_{k_0+k,i_{k+1}}) + \hat{x}$$

$\square$

B. Proof of Lemma 3

Proof. Let the initial estimate of all nodes be x_0, i.e., for node i,

$$x_{0,i} = x_0.$$

We denote $\bigtriangledown l(x_{k,i}) = \bigtriangledown l(x_{k,i}, \xi_{k,i})$ for simplicity where $\xi_{k,i}$ is the randomly selected data sample of node i at k-th iteration. Let $\mathcal{R}_{0,i}$ be the aggregated estimate for the 1st iteration using MOZI. With the GUF, we have

$$x_{1,i} = \alpha x_0 + (1 - \alpha)\mathcal{R}_{0,i} - \lambda \bigtriangledown l(x_{0,i})$$

For the 2st iteration,

$$x_{2,i} = \alpha x_1 + (1 - \alpha)\mathcal{R}_{1,i} - \lambda \bigtriangledown l(x_{1,i})$$

$$= \alpha(\alpha x_0 + (1 - \alpha)\mathcal{R}_{0,i} - \lambda \bigtriangledown l(x_{0,i})) + (1 - \alpha)\mathcal{R}_{1,i} - \lambda \bigtriangledown l(x_{1,i})$$

$$= \alpha^2 x_0 + \alpha(1 - \alpha)\mathcal{R}_{0,i} + (1 - \alpha)\mathcal{R}_{1,i} - \lambda\alpha \bigtriangledown l(x_{0,i})) - \lambda \bigtriangledown l(x_{1,i})$$

Similarly, we can represent the estimate of the k-th iteration using $\{\mathcal{R}_{ii,i}, \nabla l(x_{ii,i})\}_{ii\in[0,k-1]}$, i.e.,

$$x_{k,i} = \alpha^k x_0 + \sum_{ii=0}^{k-1} \alpha^{ii}(1-\alpha)^{k-1-ii}\mathcal{R}_{ii,i} - \lambda \sum_{ii=0}^{k-1} \alpha^{k-1-ii} \nabla l(x_{ii,i})$$

Then, for $\forall i, j \in V_b$, we have

$$||x_{k,i} - x_{k,j}|| = ||\sum_{ii=0}^{k-1} \alpha^{ii}(1-\alpha)^{k-1-ii}(\mathcal{R}_{ii,i} - \mathcal{R}_{ii,j}) - \lambda \sum_{ii=0}^{k-1} \alpha^{k-1-ii}(\nabla l(x_{ii,i}) - \nabla l(x_{ii,j}))|| \tag{15}$$

$$\leq \sum_{ii=0}^{k-1} \alpha^{ii}(1-\alpha)^{k-1-ii}||\mathcal{R}_{ii,i} - \mathcal{R}_{ii,j}|| + \lambda \sum_{ii=0}^{k-1} \alpha^{k-1-ii}|| \nabla l(x_{ii,i}) - \nabla l(x_{ii,j})||$$

Let $x_{ii,i}$, $x_{ii,j}$ be the ii-th estimates of nodes i and j, respectively. We obtain

$$||\mathcal{R}_{ii,i} - \mathcal{R}_{ii,j}|| = ||\mathcal{R}_{ii,i} - x_{ii,i} + x_{ii,i} - x_{ii,j} + x_{ii,j} - \mathcal{R}_{ii,j}|| \tag{16}$$

$$\leq ||\mathcal{R}_{ii,i} - x_{ii,i}|| + ||x_{ii,i} - x_{ii,j}|| + ||x_{ii,j} - \mathcal{R}_{ii,j}||$$

According to Algorithm 1, the nodes i and j select $\rho_i|\mathcal{N}_i|$ and $\rho_j|\mathcal{N}_j|$ neighbors, respectively. With Assumption 1, the minimal ratios of benign neighbors around i and j are ρ_i and ρ_j. Thus, there are two estimates of benign nodes, $x_{ii,i'}$ and $x_{ii,j'}$, whose neighbors are nodes i and j, respectively, such that

$$||\mathcal{R}_{ii,i} - x_{ii,i}|| \leq ||x_{ii,i'} - x_{ii,i}||,$$
$$||x_{ii,j} - \mathcal{R}_{ii,j}|| \leq ||x_{ii,j} - x_{ii,j'}||.$$

Then, under Assumption 4, we rewrite Equation 16 as

$$||\mathcal{R}_{ii,i} - \mathcal{R}_{ii,j}|| \leq ||\mathcal{R}_{ii,i} - x_{ii,i}|| + ||x_{ii,i} - x_{ii,j}|| + ||x_{ii,j} - \mathcal{R}_{ii,j}|| \tag{17}$$

$$\leq ||x_{ii,i} - x_{ii,i}|| + ||x_{ii,i} - x_{ii,j}|| + ||x_{ii,j} - x_{ii,j}||$$

$$\leq 3T$$

By combining Equation 15 and 17, we obtain

$$||x_{k,i} - x_{j,i}|| \leq 3T \sum_{ii=0}^{k-1} \alpha^{ii}(1-\alpha)^{k-1-ii} + \lambda T' \sum_{ii=0}^{k-1} \alpha^{k-1-ii}$$

$$\leq 3T \sum_{ii=0}^{k-1} \alpha^{ii} + \lambda T' \sum_{ii=0}^{k-1} \alpha^{k-1-ii}$$

$$\leq 3T \frac{1-\alpha^{k-1}}{1-\alpha} + \lambda T' \frac{1-\alpha^{k-1}}{1-\alpha}$$

$$\square$$

C. Proof of Theorem 1

Proof. Let x_{k,i^*} be the k-th estimate of node i^* in the sub-decentralized system $\mathcal{G}_b$. We also denote $\nabla l(x_{k,i}) = \nabla l(x_{k,i}, \xi_{k,i})$ for simplicity where $\xi_{k,i}$ is the randomly selected data sample of node i at k-th iteration. Then,

$$x_{k,i^*} = \alpha x_{k,i^*} + \frac{1}{|\mathcal{N}_i|} \sum_{j^* \in \mathcal{N}_{i^*}} x_{k,j^*} - \lambda \nabla l(x_{k,i^*})$$

$$= \alpha^k x_0 + \sum_{ii=0}^{k-1} \sum_{j^* \in \mathcal{N}_{i^*}} \frac{\alpha^{ii}(1-\alpha)^{k-1-ii}}{|\mathcal{N}_{i^*}|} x_{k,j^*} - \lambda \sum_{ii=0}^{k-1} \alpha^{k-1-ii} \nabla l(x_{ii,i^*})$$

where $\mathcal{N}_{i^*}$ is the neighbors of node i^* in the sub-decentralized system.

For a benign node $i \in V$, we have

$$||x_{k,i} - x_{k,i^*}|| = ||\sum_{ii=0}^{k-1} \alpha^{ii}(1-\alpha)^{k-1-ii}(\mathcal{R}_{ii,i} - \frac{1}{|\mathcal{N}_{i^*}|}\sum_{j^* \in \mathcal{N}_{i^*}} x_{k,j^*}) - \lambda \sum_{ii=0}^{k-1} \alpha^{k-1-ii}(\triangledown l(x_{ii,i}) - \triangledown l(x_{ii,i^*}))||$$

$$\leq \sum_{ii=0}^{k-1} \alpha^{ii}(1-\alpha)^{k-1-ii}||\mathcal{R}_{ii,i} - \frac{1}{|\mathcal{N}_{i^*}|}\sum_{j^* \in \mathcal{N}_{i^*}} x_{k,j^*}|| + \lambda \sum_{ii=0}^{k-1} \alpha^{k-1-ii}||\triangledown l(x_{ii,i}) - \triangledown l(x_{ii,i^*})||$$

In particular,

$$||\mathcal{R}_{ii,i} - \frac{1}{|\mathcal{N}_{i^*}|}\sum_{j^* \in \mathcal{N}_{i^*}} x_{k,j^*}|| = \frac{1}{|\mathcal{N}_{i^*}|}|||\mathcal{N}_{i^*}|\mathcal{R}_{ii,i} - \sum_{j^* \in \mathcal{N}_{i^*}} x_{k,j^*}||$$

$$= \frac{1}{|\mathcal{N}_{i^*}|}||\sum_{j^* \in \mathcal{N}_{i^*}}(\mathcal{R}_{ii,i} - x_{k,j^*})||$$

$$= \frac{1}{|\mathcal{N}_{i^*}|}||\sum_{j^* \in \mathcal{N}_{i^*}}(\mathcal{R}_{ii,i} - x_{k,j} + x_{k,j} - x_{k,j^*})||$$

where $x_{k,j}$ is the k-th estimate of the benign node j in $\mathcal{G}$ and the node j is the corresponding node of j^* in $\mathcal{G}_b$.

As shown in the proof of Lemma 3, there exists a benign node $j^* \in V$, such that

$$||\mathcal{R}_{ii,i} - x_{k,j}|| \leq ||\mathcal{R}_{ii,j^*} - x_{k,j}|| \tag{18}$$

Note that MOZI selects the estimates whose performance is better on the current training data sample. Then, with Assumption 3, a benign estimate is still benign after updating the estimate. Then, with Assumption 4, we have

$$||\mathcal{R}_{ii,i} - \frac{1}{|\mathcal{N}_{i^*}|}\sum_{j^* \in \mathcal{N}_{i^*}} x_{k,j^*}|| \leq \frac{1}{|\mathcal{N}_i|}\sum_{j \in \mathcal{N}_i}(||\mathcal{R}_{ii,i} - x_{k,j}|| + ||x_{k,j} - x_{k,j}^*||)$$

$$\leq \frac{1}{|\mathcal{N}_i|}\sum_{j \in \mathcal{N}_i}(||x_{k,j^*} - x_{k,j}|| + ||x_{k,j} - x_{k,j}^*||)$$

$$\leq 2T$$

The difference between $x_{k,i}$ and $x_{k,j}^*$ can be uniformly bounded, i.e.,

$$||x_{k,i} - x_{k,j}^*|| \leq 2T\sum_{ii=0}^{k-1}\alpha^{ii}(1-\alpha)^{k-1-ii} + \lambda T'\sum_{ii=0}^{k-1}\alpha^{k-1-ii}$$

$$\leq 2T\frac{1-\alpha^{k-1}}{1-\alpha} + \lambda T'\frac{1-\alpha^{k-1}}{1-\alpha}$$

Therefore, using Lemma 2, we obtain

$$||x_{k,i} - x^*|| \leq ||x_{k,i} - x_{k,j}^*|| + ||x_{k,i}^* - x^*||$$

$$\leq 2T\frac{1-\alpha^{k-1}}{1-\alpha} + \lambda T'\frac{1-\alpha^{k-1}}{1-\alpha} + C$$

$\square$

D. Proof of Corollary 1

Proof. Similar to the proof of Theorem 1, we denote x_{k,i^*} the k-th estimate of node i^* in the sub-decentralized system $\mathcal{G}_b$. We also denote $\triangledown l(x_{k,i}) = \triangledown l(x_{k,i}, \xi_{k,i})$ for simplicity where $\xi_{k,i}$ is the randomly selected data sample of node i at k-th iteration. Then,

$$l(x_{k+1,i}) - l(x^*) = l(\alpha x_{k,i} + (1-\alpha)\mathcal{R}_{k,i} - \lambda \triangledown l(x_{k,i})) - l(x^*)$$

According to our assumption, l is a convex function, we have

$$l(x_{k+1,i}) - l(x^*) \leq \alpha l(x_{k,i} - \lambda \triangledown l(x_{k,i})) + (1-\alpha)l(\mathcal{R}_{k,i} - \lambda \triangledown l(x_{k,i})) - l(x^*) \tag{19}$$

At each iteration, MOZI selects potential benign estimates according to the estimates and loss of a benign node. If $l(x_{k,i}) - l(x_{k,j}) \geq 0, \forall j \in \mathcal{N}_{k,i}^r$, we obtain

$$l(\mathcal{R}_{k,i} - \lambda \triangledown l(x_{k,i})) \leq l(x_{k+1,i}) - l(x^*) \leq \alpha l(x_{k,i} - \lambda \triangledown l(x_{k,i})) \tag{20}$$

Thus, with Lemma 2, we have

$$l(x_{k+1,i}) - l(x^*) \leq l(x_{k,i} - \lambda \bigtriangledown l(x_{k,i})) - l(x^*)$$
$$\leq C'$$

Otherwise, $\mathcal{R}_{k,i} = x_{k,j^*}$, $j^* = \mathrm{argmin}_{j \in \mathcal{N}_{k,i}^s} l_{k,j}$. Here, we assume that there is at least one benign estimate in $\mathcal{N}_{k,i}^s$. With Assumption 2, we have

$$l(x_{k+1,i}) - l(x^*) \leq \alpha(l(x_{k,i} - \lambda \bigtriangledown l(x_{k,i})) - l(x^*)) + (1 - \alpha)(l(\mathcal{R}_{k,i} - \lambda \bigtriangledown l(x_{k,i})) - l(x^*))$$
$$\leq C'$$

$\square$

E. Experiments on MNIST

(a) 30 nodes   (b) 50 nodes   (c) 70 nodes

Fig. 8. MNIST: The worst accuracy of the benign nodes with different network sizes.

(a) Connection ratio = 0.2   (b) Connection ratio = 0.4   (c) Connection ratio = 0.6

Fig. 9. MNIST: The worst accuracy of the benign nodes with different connection ratios.

(a) Byzantine ratio = 0.1   (b) Byzantine ratio = 0.3   (c) Byzantine ratio = 0.5

Fig. 10. MNIST: the worst accuracy of the benign nodes under the Gaussian attack.

(a) Byzantine ratio = 0.1   (b) Byzantine ratio = 0.3   (c) Byzantine ratio = 0.5

Fig. 11. MNIST: the worst accuracy of the benign nodes under the bit-flip attack.

(a) Byzantine ratio = 0.1   (b) Byzantine ratio = 0.3   (c) Byzantine ratio = 0.5
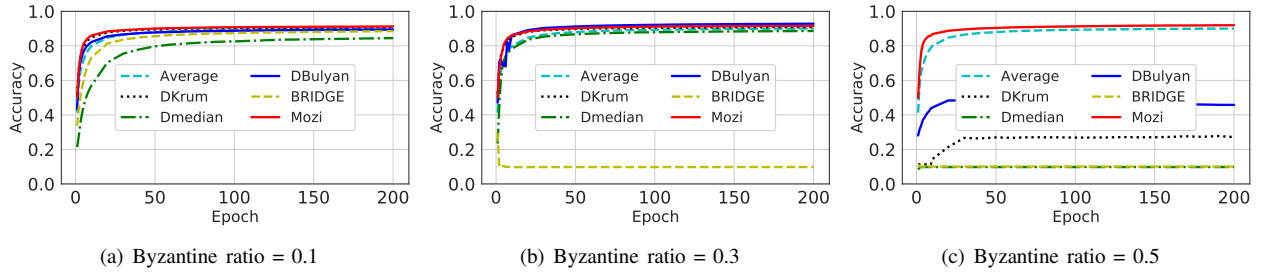
Fig. 12. MNIST: the worst accuracy of the benign nodes under the Mhamdi attack.